



This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출원 번호 : 10-2002-0069658  
Application Number PATENT-2002-0069658

출원 년 월 일 : 2002년 11월 11일  
Date of Application NOV 11, 2002

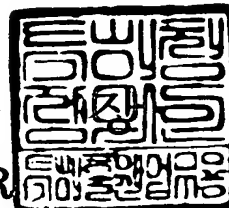
출원 인 : 삼성전자주식회사  
Applicant(s) SAMSUNG ELECTRONICS CO., LTD.



2003 년 01 월 28 일

특 허 청

COMMISSIONER



## 【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0022
【제출일자】	2002.11.11
【국제특허분류】	G06F
【발명의 명칭】	메모리를 감소시키는 개선된 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법
【발명의 영문명칭】	The improved method of compressing look up table for reducing memory and non-linear function generating apparatus having look up table compressed using the method and the non-linear function generating method
【출원인】	
【명칭】	삼성전자 주식회사
【출원인코드】	1-1998-104271-3
【대리인】	
【성명】	이영필
【대리인코드】	9-1998-000334-6
【포괄위임등록번호】	1999-009556-9
【대리인】	
【성명】	정상빈
【대리인코드】	9-1998-000541-1
【포괄위임등록번호】	1999-009617-5
【발명자】	
【성명의 국문표기】	이헌수
【성명의 영문표기】	LEE, Heon Soo
【주민등록번호】	731214-1622518
【우편번호】	449-711
【주소】	경기도 용인시 기흥읍 삼성전자(주)기흥공장 남자기숙사 상록수동 40 5호
【국적】	KR
【심사청구】	청구

## 【취지】

특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사를 청구합니다. 대리인

이영필 (인) 대리인

정상빈 (인)

## 【수수료】

【기본출원료】 20 면 29,000 원

【가산출원료】 19 면 19,000 원

【우선권주장료】 0 건 0 원

【심사청구료】 29 항 1,037,000 원

【합계】 1,085,000 원

## 【첨부서류】

1. 요약서·명세서(도면)\_1통

## 【요약서】

## 【요약】

메모리를 감소시키는 개선된 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법이 개시된다. 본 발명에 의한 메모리를 감소시키는 개선된 룩업 테이블 압축방법은, 기울기 변화가 감소하는 비선형 함수 발생장치의 룩업 테이블 압축방법에 있어서, (a) 비선형 함수의 X좌표들을 소정의 좌표 간격을 가지는 스텝을 각각 포함하는 복수개의 구간들로 분할하는 단계; (b) 상기 각 스텝별 X좌표 값에 대응하는 Y좌표 값을 추출하는 단계; 및 (c) 상기 각 구간의 스텝별 X좌표 값 및 상기 Y좌표 값들을 메모리 내의 지정된 주소에 저장하는 단계를 포함하며, 스텝의 좌표 간격은 복수개의 구간들에 대하여 다르게 설정되는 것을 특징으로 한다.

본 발명에 의한 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법에 의하면, 룩업 테이블이 차지하는 메모리의 용량을 줄일 수 있는 장점이 있다.

## 【대표도】

도 5

**【명세서】****【발명의 명칭】**

메모리를 감소시키는 개선된 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법{The improved method of compressing look up table for reducing memory and non-linear function generating apparatus having look up table compressed using the method and the non-linear function generating method}

**【도면의 간단한 설명】**

도 1은 종래 기술에 따른 비선형 함수 발생장치의 일예를 개략적으로 나타내는 블록도이다.

도 2는 종래 기술에 따른 비선형 함수 발생장치의 룩업 테이블 압축과정을 설명하기 위한 그래프 도이다.

도 3은 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축과정을 설명하기 위한 그래프 도이다.

도 4는 도 3에 도시된 그래프의 주요 데이터들과 룩업 테이블을 나타내는 도면이다

도 5는 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축과정을 나타내는 플로우차트이다.

도 6은 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축 방법을 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치를 개략적으로 나타내는 블록도이다.

도 7은 도 6에 도시된 비선형 함수 발생장치의 비선형 함수 발생과정을 나타내는 플로우차트이다.

도 8은 도 6에 도시된 비선형 함수 발생장치의 근사값 연산에 사용되는 X좌표 값들 및 Y좌표 값들의 관계를 나타내는 그래프 도이다.

도 9는 도 7에 도시된 플로우차트에서 구간 정보 및 스텝 정보를 산출하는 과정을 상세히 나타내는 플로우차트이다.

도 10은 도 7에 도시된 플로우차트에서 X좌표에 근접하는 룩업 테이블 상의 X좌표 값을 산출하는 과정을 상세히 나타내는 플로우차트이다.

#### 【발명의 상세한 설명】

#### 【발명의 목적】

#### 【발명이 속하는 기술분야 및 그 분야의 종래기술】

<11> 본 발명은 디지털 신호처리에 사용되는 비선형 함수 발생장치의 룩업 테이블 압축 방법 및 비선형 함수 발생장치 및 그 발생방법에 관한 것으로서, 특히, 메모리를 감소시키는 개선된 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법에 관한 것이다.

<12> 일반적으로, 디지털 신호처리 반도체 장치에서는 비선형 함수가 사용되는 경우가 많다. 이러한 비선형 함수를 구현하기 위해서 수식이나 룩업 테이블(함수표, look up

table)이 사용된다. 수식을 사용하여 비선형 함수를 구현하는데는 많은 계산량이 요구되므로, 더 간단한 룩업 테이블이 자주 사용된다. 그러나, 룩업 테이블은 많은 양의 메모리를 필요로 하기 때문에 작은 크기의 룩업 테이블을 이용하면서도 더 정확한 근사값을 구하기 위한 여러 방법들이 제안되고 있다. 룩업 테이블의 크기를 축소시킨 비선형 함수 발생장치의 일례로서, 국내 특허 등록 제219543호(축소된 룩업 테이블을 이용한 비선형 함수 발생장치)가 있다.

<13> 국내 특허 등록 제219543호는 CRT의 영상신호 감마보정에 적용되는 비선형 함수 발생장치에 관한 것이다. 국내 특허 등록 제219543호의 축소된 룩업 테이블을 이용한 비선형 함수 발생장치가 도 1에 도시되며, 도 1을 참조하여 상세히 설명하면 다음과 같다.

<14> 도 1은 종래 기술에 따른 비선형 함수 발생장치의 일례를 개략적으로 나타내는 블록도이다.

<15> 도 1과 같이, 종래 기술에 따른 비선형 함수 발생장치는 제1 룩업 테이블(11), 제2 룩업 테이블(12), 제1 가산기(13), 승산기(14) 및 제2 가산기(15)를 구비한다.

<16> 상기 제1 룩업 테이블(11)에는 기준 입력과 그에 대응하는 출력에 대해 출력과 입력의 차가 저장되고, 상기 제2 룩업 테이블(12)에는 기준 입력과 인접한 다른 기준 입력들과 그들에 각각 대응하는 출력들로부터의 기울기가 저장된다.

<17> 상기 제1 및 제2 가산기(13, 15)와, 상기 승산기(14)는 상기 기준 입력과 상기 제1 및 제2 룩업 테이블(11, 12)로부터의 출력신호들을 연산하여, 상기 기준 입력에 대응하는 근사값을 출력한다.

- <18>      상기와 같이, 종래의 비선형 함수 발생장치의 상기 제1 및 제2 룩업 테이블(11, 12)에 비선형 함수가 등간격으로 분할되어 추출된 기준 값들 및 기울기 값들만이 저장된다. 따라서, 룩업 테이블이 차지하는 메모리의 용량이 감소될 수 있다.
- <19>      한편, 이러한 비선형 함수가 적용되는 일례로서, MP3 디코더의 디코딩 과정을 들 수 있다. MP3 디코더의 디코딩 과정을 간략히 살펴보면 다음과 같다.
- <20>      먼저, MP3 파일의 비트 스트림으로부터 프레임 헤더(frame header)를 찾는다. 이 후, 프레임의 인코딩 정보인 사이드 인포메이션(side information)과, 스케일 팩터(scale factor) 및 허프만 코딩 데이터를 차례로 분리한다. 다음으로 상기 사이드 인포메이션에 근거하여 허프만 디코딩이 수행되고, 역양자화 스펙트럼(requantize spectrum) 과정에 의해 상기 허프만 디코딩 결과를 주파수 영역에서의 실제 샘플 에너지 값으로 복원한다.
- <21>      이 후, 리오더링(reordering) 과정, 스테레오 디코딩(stereo decoding) 과정, 에일리어스(alias) 감소 과정, IMDCT 및 오버랩(overlap) 합산 과정 등이 수행되어 PCM 데이터가 출력되고, 상기 PCM 데이터는 D/A 컨버터에 의해 아날로그 신호로 변환되어 출력된다.
- <22>      상기와 같은 MP3 디코더의 디코딩 과정들 중에서 비선형 함수가 사용되는 과정은 상기 역양자화 스펙트럼 과정이다. 상기 역양자화 스펙트럼 과정에서는 허프만 디코딩된 결과를 주파수 영역에서 실제 샘플 값으로 복원하는 작업을 수행하게 되는데, 이 때 사용되는 비선형 함수가  $Y = X^{4/3}$  (단,  $X = 0, 1/8192, 2/8192, 3/8192, \dots, 8191/8192$ ) 함수이다. 종래의 룩업 테이블 압축 방법에서는  $Y = X^{4/3}$  함수의 값이 등간격으로 추출된다. 이를 도 2에 도시된 그래프를 참고하여 좀 더 상세히 설명하면 다음과 같다.

- <23> 도 2는 종래 기술에 따른 비선형 함수 발생장치의 록업 테이블 압축과정을 설명하기 위한 그래프 도이다.
- <24> 도 2와 같이,  $Y = X^{4/3}$  함수의 X좌표들이 동일한 간격을 갖는 복수개의 구간들(P1 ~ P6)로 분할되고, 각 구간(P1~P6)별로 추출된 기준 값들과, 상기 등간격의 기준 값들 사이의 값들을 계산하기 위해 각 구간(P1~P6)에 대한 기울기 값이 저장된다.
- <25> 도 2에서,  $Y = X^{4/3}$  함수의 기울기 변화를 살펴보면, 초기에는 기울기가 급격히 변하면서 곡선의 형태를 띄고, 이 후 X좌표 값이 증가될 수록 기울기의 변화가 일정하게 감소되는 것을 알 수 있다.
- <26> 그러나, 종래 기술에 따른 비선형 함수 발생장치의 록업 테이블 압축과정에서는 함수 값이 등간격으로 추출되어 곡선부분에서의 급격히 변하는 기울기를 모두 만족시킬 수 없으므로 오차가 발생된다.
- <27> 기울기의 변화가 일정하게 감소되는 비선형 함수에 대하여, 종래의 방법은 기울기가 급격히 변화되는 구간(P1 ~ P3)에서는 많은 오차가 발생되고, 기울기의 변화가 일정하게 감소되는 구간(P4 ~ P6)에서는 불필요하게 많은 데이터들이 저장되어 비효율적인 것이다.
- <28> 따라서, 2차 이상의 다항식의 단조 증감 구간,  $y = e^{-x}$ 와 같은 감소하는 지수 함수, 로그 함수, 감마보정 스케일 함수 등과 같이 기울기 변화가 일정하게 감소하는 함수들을 위한 개선된 록업 테이블 압축방법이 절실히 요구된다.

**【발명이 이루고자 하는 기술적 과제】**

<29> 본 발명이 이루고자하는 기술적 과제는, 비선형 함수가 복수개의 구간으로 분할 될 때, 기울기 변화가 감소되는 구간일 수록 그 분할 간격을 증가시켜 함수 값을 추출함으로써, 메모리를 감소시키는 개선된 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법을 제공하는데 있다.

**【발명의 구성 및 작용】**

<30> 상기 기술적 과제를 달성하기 위한 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축방법은, 기울기 변화가 감소하는 비선형 함수 발생장치의 룩업 테이블 압축방법에 있어서,

<31> (a) 비선형 함수의 X좌표들을 소정의 좌표 간격을 가지는 스텝을 각각 포함하는 복수개의 구간들로 분할하는 단계;

<32> (b) 상기 각 스텝별 X좌표 값에 대응하는 Y좌표 값을 추출하는 단계; 및

<33> (c) 상기 Y좌표 값들을 메모리 내의 지정된 주소에 저장하는 단계를 포함하며,

<34> 상기 스텝의 상기 좌표 간격은 상기 복수개의 구간들에 대하여 다르게 설정되는 것을 특징으로 한다.

<35> 상기 기술적 과제를 달성하기 위한 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블을 가지는 비선형 함수 발생장치는, 기울기 변화가 감소하는 비선형 함수를 발생시키는 장치에 있어서, 분석기, 메모리 및 연산기를 구비하는 것을 특징으로 한다.

<36> 분석기는 입력되는 X좌표 값으로부터 어드레스를 포함하는 분석정보를 산출하여 출력한다. 메모리는 소정의 좌표 간격을 가지는 스텝을 포함하는 복수개의 구간들로 분할되는 비선형 함수의 스텝별 X좌표 값에 대응하는 Y좌표 값이 지정된 주소에 저장되는 룩업 테이블을 포함하며, 어드레스에 대응하는 Y좌표 값을 출력한다. 연산기는 분석정보 및 어드레스에 대응하는 Y좌표 값들로부터 입력되는 X좌표 값에 대응하는 Y좌표의 근사 값을 계산하여 출력한다. 스텝의 좌표 간격은 복수개의 구간들에 대하여 다르게 설정된다.

<37> 상기 기술적 과제를 달성하기 위한 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블을 가지는 비선형 함수 발생장치의 비선형 함수 발생방법은, 기울기 변화가 감소하고, 소정의 좌표 간격을 가지는 스텝을 포함하는 복수개의 구간들로 분할되는 비선형 함수의 상기 스텝별 X좌표 값에 대응하는 Y좌표 값이 지정된 주소에 저장되는 룩업 테이블을 포함하는 비선형 함수 발생장치의 비선형 함수 발생방법에 있어서,

<38> (a) 입력되는 X좌표 값으로부터 구간 정보 및 스텝 정보를 산출하는 단계;

<39> (b) 상기 구간 정보 및 스텝 정보로부터 어드레스를 산출하는 단계;

<40> (c) 상기 룩업 테이블로부터 상기 어드레스에 대응하는 Y좌표를 획득하는 단계;

<41> (d) 상기 어드레스로부터 입력되는 X좌표의 근접 X좌표 값을 산출하는 단계;

<42> (e) 상기 구간정보로부터 해당 구간에 포함되는 스텝의 좌표 간격 값을 산출하는 단계; 및

- <43> (f) 상기 산출된 X좌표의 근접 X좌표 값, 상기 어드레스에 대응하는 Y좌표 값 및 상기 좌표 간격 값으로부터 상기 입력 X좌표 값에 대응하는 Y좌표의 근사값을 계산하여 출력하는 단계를 포함하는 것을 특징으로 한다.
- <44> 본 발명과 본 발명의 동작상의 이점 및 본 발명의 실시에 의하여 달성되는 목적을 충분히 이해하기 위해서는 본 발명의 바람직한 실시예를 예시하는 첨부 도면 및 도면에 기재된 내용을 참조하여야 한다.
- <45> 이하, 첨부한 도면을 참조하여 본 발명의 바람직한 실시예를 설명함으로써, 본 발명을 상세히 설명한다. 각 도면에 제시된 동일한 참조부호는 동일한 부재를 나타낸다.
- <46> 먼저, 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축 과정을 도 3 내지 도 5를 참조하여 설명하면 다음과 같다.
- <47> 도 3은 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축 과정을 설명하기 위한 그래프 도이고, 도 4는 도 3에 도시된 그래프의 주요 데이터들과 룩업 테이블을 나타내는 도면이다. 또, 도 5는 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축 과정을 나타내는 플로우차트(100)이다.
- <48> 먼저, 도 3을 참고하면, 비선형 함수  $Y = X^{4/3}$ 의 X좌표가 복수개의 구간들( $i_0 \sim i_4$ )로 분할된다. 또, 상기 복수개의 구간들( $i_0 \sim i_4$ )은 소정의 좌표 간격을 갖는 4개의 스텝들( $j_0 \sim j_3$ )을 포함한다.
- <49> 상기 비선형 함수  $Y = X^{4/3}$ 의 기울기가 급격히 변하는 곡선 구간인 구간들( $i_0, i_1$ )은 상기 스텝들( $j_0 \sim j_3$ )간의 좌표 간격이 "1"이고, 상기 비선형 함수  $Y = X^{4/3}$ 의 기울기 변화가 감소되는 직선 구간인 구간들( $i_2, i_3, i_4$ )은 상기 스텝들( $j_0 \sim j_3$ )간의 간격이 점

점 증가된다. 이를 좀 더 상세히 설명하면, 상기 구간( $i_2$ ) 내의 스텝들( $j_0 \sim j_3$ )간의 좌표 간격은 "2"이고, 상기 구간( $i_3$ ) 내의 스텝들( $j_0 \sim j_3$ )간의 좌표 간격은 "4"이다. 또, 상기 구간( $i_4$ ) 내의 스텝들( $j_0 \sim j_3$ )간의 좌표 간격은 "8"이다.

<50> 결국, 상기 비선형 함수  $Y = X^{4/3}$ 의 기울기 변화가 감소되는 구간에서 스텝들간의 좌표 간격이 2의 지수승으로 증가되는 것을 알 수 있다. 단, 예외적으로 상기 비선형 함수  $Y = X^{4/3}$ 의 기울기가 급격히 변화되는 곡선 구간에서 스텝들간의 좌표 간격은 동일하게 설정된다. 그 이유는 급격한 기울기 변화에 따른 오차를 감소시키기 위함이다.

<51> 도 3에 도시된 비선형 함수  $Y = X^{4/3}$ 의 X좌표 값들 및 Y좌표 값들과 구간 및 스텝들을 표로 나타내면 도 4와 같다.

<52> 도 4에서, X 좌표들은 복수개의 구간들( $i$ )로 분할되며, 각 구간들은 복수개의 스텝들( $j_0, j_1, j_2, j_3$ )을 포함한다. 여기에서, 상기 복수개의 구간들은 구간순서( $i_0$ )와 구간번호( $i_N$ )가 다르게 설정된다.

<53> 상기 구간순서( $i_0$ )는 " $i_0, i_1, i_2, i_3, i_4, \dots$ "순으로 증가되고, 상기 구간번호( $i_N$ )는 상기 첫 번째 구간( $i_0$ ) 및 상기 두 번째 구간( $i_1$ )이 "0"이고, 상기 세 번째 구간( $i_2$ )부터 "1, 2, 3, ..."순으로 증가된다.

<54> 또, 도 4에서 X좌표를 이진수로 나타낸 부분을 살펴보면, 상기 복수개의 구간들( $i_0, i_1, i_2, i_3, i_4$ ) 모두 상기 X좌표 값이 각 스텝별로 변화 비트들(A), 즉, "00", "01", "10", "11"을 포함한다.

<55> 또, 첫 번째 구간( $i_0$ )을 제외한 나머지 구간들( $i_1, i_2, i_3, i_4$ )의 X좌표 값들은 상기 스텝별 변화 비트들(A)의 상위 비트로서 "1"을 더 포함한다.

<56> 도 4에서, 상기 X좌표 값에 대응하는 Y좌표 값들이 룩업 테이블(41)로서 메모리(도 6의 40 참조)에 저장된다.

<57> 도 3 및 도 4에서는 한 구간 내에 포함되는 스텝의 개수가 4개이고, 각 구간별 스텝들간의 좌표 간격이 2의 지수승으로 증가되는 것을 예를 들어 설명하였으나, 상기 스텝의 개수는 다양하게 설정될 수 있다. 또, 한 구간 내에 포함되는 스텝의 개수에 따라 정밀도가 달라지게 된다. 한 구간 내에 포함되는 스텝의 개수와 룩업 테이블의 크기와의 관계를 표로 나타내면 다음과 같다.

<58> 【표 1】

구간의 스텝수	구간의 총수	룩업 테이블의 크기 (항목)
1	14	14
2	13	26
3	12	48
4	11	88
16	10	160
32	9	288
64	8	512

<59> 상기 [표 1]은 비선형 함수  $Y = X^{4/3}$ 를 "0~8191"의 X좌표에 대응하는 Y좌표 값을 표현하고자 할 때의 표의 크기를 나타내며, 한 구간의 스텝 수에 따라 구간의 수가 결정된다. 상기 스텝 수와 구간의 수는 서로 반비례하며, 한 구간의 스텝 수를 크게 할 수록 표의 크기는 증가되지만, 오차는 더 감소된다.

<60> 상기 [표 1]과 같이, 본 발명의 일실시예에 따른 룩업 테이블 압축방법에 의하면, 압축하지 않은 경우(8192 항목)에 비해 룩업 테이블의 크기가 현저하게 감소되는 것을

알 수 있다. 또, 본 발명에 의한 비선형 함수 발생장치는 별도의 기울기표를 필요로 하지 않기 때문에, 상기 룩업 테이블이 차지하는 메모리 용량을 좀 더 줄일 수 있다.

<61>       또, 도 4에서 X좌표 값이 7비트의 이진수로 표현되었으나, 여기에 한정되는 것은 아니며 입력되는 X좌표 값의 범위에 따라 다양하게 변경될 수 있다. 예를 들어, 입력되는 X좌표 값이 "0~8191"의 값이라면 13비트로 표현될 수 있다.

<62>       다음으로, 도 5의 플로우차트(100)를 참고하여 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축 과정을 설명하면 다음과 같다. 상기 플로우차트(100)는 다음의 과정들로 수행된다.

<63>       먼저, 도 3과 같이, 비선형 함수의 X좌표들이 소정의 좌표 간격을 가지는 스텝들 ( $j_0, j_1, j_2, j_3$ )을 각각 포함하는 복수개의 구간들( $i_0, i_1, i_2, i_3, i_4, \dots$ )로 분할된다(101). 이 후, 상기 각 스텝별 X좌표 값에 대응하는 Y좌표 값이 추출된다(102). 다음으로, 상기 Y좌표 값들이 메모리 내의 지정된 주소에 저장되어 룩업 테이블이 형성된다(103).

<64>       도 6은 본 발명의 일실시예에 따른 메모리를 감소시키는 개선된 룩업 테이블 압축 방법을 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치를 개략적으로 나타내는 블록도이다.

<65>       도 6과 같이, 본 발명의 일실시예에 따른 비선형 함수 발생장치(20)는 분석기(30), 메모리(40) 및 연산기(50)를 구비한다.

<66>       상기 분석기(30)는 입력되는 X좌표로부터 어드레스를 포함하는 분석정보를 산출하여 출력한다. 상기 분석정보는 복수개의 구간 중 상기 입력되는 X좌표가 속하는 구간과

스텝의 구간정보 및 스텝정보와, 상기 스텝들간의 좌표 간격(STEP SIZE), 상기 입력되는 X의 근접 X좌표 값을 포함한다.

<67> 상기 근접 X좌표 값은 룩업 테이블에 존재하는 항목들에 해당하는 X좌표 값들 중에서 상기 입력되는 X좌표 값 보다 크지 않은 최대값을 의미하며, 이하, 설명의 편의상  $X_0$ 라 한다.

<68> 여기에서, 상기 구간정보는 구간번호( $i_N$ )이고, 상기 어드레스는 상기 구간정보 및 상기 스텝정보로부터 산출된다.

<69> 상기 메모리(40)는 룩업 테이블(도 4의 41 참조)을 포함하며, 상기 어드레스에 대응하는 Y좌표 값을 출력한다. 상기 룩업 테이블은 소정의 스텝별 좌표 간격으로 추출된 복수개의 Y좌표 값이 지정된 주소에 저장된다.

<70> 상기 연산기(50)는 상기 분석정보 및 상기 Y좌표 값으로부터 상기 입력 X좌표에 대응하는 Y좌표의 근사값을 계산하여 출력한다.

<71> 상기와 같이 구성된 비선형 함수 발생장치(20)의 동작을 도 7을 참고하여 설명하면 다음과 같다.

<72> 도 7은 도 6에 도시된 비선형 함수 발생장치의 비선형 함수 발생과정을 나타내는 플로우차트(1000)이다. 상기 플로우차트(1000)는 다음의 과정들로 수행된다.

<73> 도 7과 같이, 먼저, 분석기(30)가 입력되는 X좌표 값으로부터 구간 정보 및 스텝 정보를 산출하고(1100), 상기 구간 정보 및 스텝 정보로부터 어드레스를 산출하여 메모리(22)로 출력한다(1200).

- <74>        여기에서, 상기 구간 정보는 구간 번호( $i_N$ )이고, 상기 스텝 정보는 스텝별 변화 비트(A)인 하위 2비트와, 상기 하위 2비트의 앞에 위치한 1비트를 포함하는 3비트 데이터이다. 도 4를 참고하면, 두 번째 구간( $i_1$ )의 세 번째 스텝( $j_2$ )인 경우, 상기 스텝 정보는 "110"이 된다. 상기 단계(1100)에 대해서는 도 9를 참조하여 좀 더 상세히 후술된다.
- <75>        상기 어드레스는 상기 구간 번호( $i_N$ )를 좌측으로 두 번 쉬프트시킨 후 상기 스텝 정보를 가산하는 것에 의해 얻어진다. 상기와 같이, 상기 구간 번호( $i_N$ )를 좌측으로 두 번 쉬프트시키면, 상기 구간 번호( $i_N$ )에 4를 곱한 것과 동일한 효과를 얻는다. 여기에서, 곱셈기를 사용하지 않는 이유는 반도체 장치에서 곱셈기가 차지하는 크기가 크기 때문이며, 상기와 같은 방법으로 곱셈기의 기능을 대신하게 된다.
- <76>        또, 상기 구간 번호( $i_N$ )에 4를 곱하는 이유는 한 구간이 4개의 스텝들을 포함하기 때문이다.
- <77>        좀 더 상세히 설명하면, 예를 들어, 상기 구간 번호( $i_N$ )가 2("10")이고, 상기 스텝 정보( $j$ )가 "101"인 경우 어드레스를 계산하면 다음과 같다.
- <78>        먼저, 상기 구간 번호( $i_N$ ) "10"을 좌측으로 두 번 쉬프트시키면 "1000"이 되고, 여기에 상기 스텝 정보( $j$ )인 "101"을 가산하면, "1101"이 된다. 결국, 어드레스 13("1101")이 얻어진다.
- <79>        이 후, 상기 메모리(40) 내의 룩업 테이블(41)로부터 상기 어드레스에 대응하는 Y좌표 값이 출력된다(1300). 여기에서, 상기 메모리(40)로부터 두 개의 Y좌표 값 ( $table[index], table[index + 1]$ )(도 6 참조)이 출력되며, 상기 두 개의 Y좌표 값은 근

사값 계산에 필요한 값들로서, 상기 X좌표 값이 속하는 스텝들에 대한 어드레스들 (index, index + 1)에 대응하는 Y좌표 값들이다.

- <80> 또, 상기 분석기(30)에 의해 상기 어드레스로부터 X의 X0 값(도 6 참고)이 산출된다(1400). 상기 단계(1400)에 대해서는 도 10을 참조하여 좀 더 상세히 후술된다.
- <81> 상기 분석기(30)는 단계(100)에서 구한 상기 구간 정보로부터 해당 구간에 포함되는 스텝들간의 좌표 간격을 산출한다(1500).
- <82> 여기에서, 상기 단계(1500)에서 스텝들간의 좌표 간격을 산출하는 과정을 C 언어로 표현하면 다음의 표와 같다.

<83> 【표 2】

```
int x2step(int X){
    int i;
    int j = x;
    for(i=0;j>=8;i++)
        j >>= 1;          /* 구간 번호(iN)를 구함 */
    return 1 << i;         /* 2iN이 스텝들간의 좌표 간격이 됨 */
}
```

- <84> 여기에서, 상기 i는 구간 번호( $i_N$ )를 나타내고, j는 구간내의 스텝의 위치를 나타낸다.
- <85> 상기와 같이, 스텝들간의 좌표 간격은  $2^{i_N}$ 으로 계산된다. 따라서, 예를 들어, 구간 번호( $i_N$ )가 "0"인 경우 스텝들간의 좌표 간격은  $2^0 = 1$ 이 되고, 구간 번호( $i_N$ )가 "2"인 경우 스텝의 좌표 간격은  $2^2 = 4$ 가 된다.

- <86>      상기 연산기(50)는 상기 X의  $X_0$  값, 상기 Y좌표 값 및 상기 스텝들간의 좌표 간격으로부터 상기 X좌표 값에 대응하는 Y좌표 값의 근사값을 계산하여 출력한다(1600).
- <87>      여기에서, 상기 근사값은 다음의 선형 함수  $Y = aX + b$ (도 8 참고)(a : 기울기, b :  $X_0$ 에 대응하는 Y좌표 값)에 의해 계산된다.
- <88>      근사값( $Y$ ) =  $\text{table}[\text{index}] + (\text{table}[\text{index}+1] - \text{table}[\text{index}])(X - X_0)/\text{STEP SIZE}$
- <89>      여기에서, 상기 b는  $\text{table}[\text{index}]$ 이고, 상기 a는  $(\text{table}[\text{index}+1] - \text{table}[\text{index}])/\text{STEP SIZE}$  이고, 상기  $(X - X_0)$ 는 X의 변위 값이다. 또, 상기 STEP SIZE는 스텝의 좌표 간격을 나타내고, 상기  $X_0$ 는 X를 인덱스(index) 함수에 의해 인덱스 값으로 변환한 후, 이것을 다시 X좌표 값으로 변환한 값이다. 상기와 같이 X를 인덱스 값에서 X좌표 값으로 변환하면, 상기 X의 가장 근접한 인덱스에 해당되는 근접 X좌표 값( $X_0$ )이 얻어진다. 여기에서, 상기 인덱스 함수는 상기 분석기(30)(도 6 참고)에 포함된다.
- <90>      도 8은 도 6에 도시된 비선형 함수 발생장치의 근사값 연산에 사용되는 X좌표 값들 및 Y좌표 값들의 관계를 나타내는 그래프 도이다.
- <91>      도 8과 같이, X에 대한 함수( $Y = X^{4/3}$ )의 근사값은 선형 함수( $Y = aX + b$ )로부터 계산된다. 도 8에서, " $\text{table}[\text{index}]$ "과, " $\text{table}[\text{index} + 1]$ "은 분석기(30)로부터 출력된 어드레스에 의해 룩업 테이블로부터 출력된 Y좌표 값들이다.
- <92>      " $X_0$ "는 X를 인덱스 함수에 의해 인덱스 값으로 변환한 후, 이것을 다시 X좌표 값으로 변환한 값이다.

- <93> 도 9는 도 7에 도시된 플로우차트에서 구간 정보 및 스텝 정보를 산출하는 과정을 상세히 나타내는 플로우차트(1100)이다. 상기 플로우차트(1100)는 다음의 과정들로 수행된다.
- <94> 도 9와 같이, 먼저, 소정의 비트들을 가지는 X좌표 값을 수신하면(1101), 상기 X좌표 값이 소정 값 미만이 될 때까지 상기 X좌표 값의 각 비트들을 우측으로 한 비트씩 쉬프트 시킨다(1102, 1103). 상기 소정 값은 한 구간 내의 스텝의 개수  $\times 2$ 의 값이 된다. 도 9에서는 한 구간 내의 스텝의 개수가 4개인 것을 예로서 설명한다. 따라서, 상기 소정 값은 "8(1000)"이 된다.
- <95> 또, 상기 X좌표 값을 쉬프트시킬 때마다 그 횟수를 카운팅 한다(1104). 이 후, 상기 X좌표 값이 상기 "8(1000)" 미만일 때, 상기 카운팅된 쉬프팅 횟수로부터 상기 구간 정보를 얻는다(1105). 여기에서, 상기 구간정보는 구간번호( $i_N$ )이다.
- <96> 다음으로, 상기 쉬프트된 X좌표 값으로부터 상기 스텝 정보를 얻는다(1106). 여기에서, 상기 스텝 정보는 스텝별 변화 비트(A)인 하위 2비트와, 상기 하위 2비트의 앞에 위치한 1비트를 포함하는 3비트 데이터이다.
- <97> 상기 구간 정보 및 스텝 정보의 산출 과정을 C 언어로 표현하면 아래의 표와 같다.
- <98> 【표 3】

```

int x2index(int x) {
    int i;
    int j = x;
    for(i=0; j>=8; i++)
        j>>=1; /* 우측 쉬프트 1은 나누기 2와 동일한 효과 */
    return(i<<2) + j; /* 좌측 쉬프트 2는 곱하기 4와 동일한 효과 */
}

```

- <99>        여기에서, 상기  $i$ 는 구간 번호( $i_N$ )를 나타내고,  $j$ 는 구간내의 스텝의 위치를 나타낸다. 또, 상기 "return" 값은 어드레스를 나타낸다.
- <100>        상기 구간 정보 및 스텝 정보의 산출 과정을 일예를 들어 좀 더 상세히 설명하면 다음과 같다.
- <101>        X좌표 값이 42(0101010)가 입력될 경우, X좌표 값이 8(1000)이상이면 우측으로 한 비트씩 반복적으로 쉬프트된다. 상기 X좌표 값 42가 쉬프트 되는 과정을 나타내면 아래의 표와 같다.
- <102>        【표 4】

0101010	$i = 0$
0010101	$i = 1$
0001010	$i = 2$
0000101	$i = 3$

- <103>        상기 쉬프트 결과로부터, 쉬프트 횟수가 3이므로, 구간 번호( $i_N$ )는 3인 것을 알 수 있다. 또, 최종 쉬프트 결과 값이 "101" 이므로, 하위 2비트인 "01"을 변화 비트들(A)과 비교하면 두 번째 스텝임을 알 수 있다. 따라서, 구간 정보 및 스텝 정보는 도 4의 표로부터  $i_4$  구간의 두 번째 스텝( $j_1$ )이 된다.
- <104>        또, 여기에서 상기 구간 정보 및 스텝 정보를 이용하여 어드레스를 산출하기 위해서, 상기 구간 정보를 좌측으로 두 번 쉬프트 한 후, 상기 스텝 정보를 가산한다.

- <105> 먼저, 상기 구간 정보 "3(0011)"를 좌측으로 두 번 쉬프트 하면, "1100"이 되고, 여기에 상기 스텝 정보 "101"을 가산하면, 상기 어드레스는 "10001(17)"이 된다.
- <106> 도 10은 도 7에 도시된 플로우차트에서 X0 값을 산출하는 과정을 상세히 나타내는 플로우차트(1400)이다.
- <107> 도 10에서, 먼저, 소정의 비트들을 가지는 어드레스의 하위 2비트를 제외한 나머지 비트들을 마스킹 한다(1401). 여기에서, 마스킹은 상기 어드레스에 "3(0000011)"을 AND 연산하는 것에 의해 수행될 수 있다.
- <108> 상기 어드레스의 마스킹 되지 않은 하위 2비트들로부터 변화 비트들(A)을 얻는다(1402). 이 후, 상기 어드레스의 하위 2비트들을 마스킹한다(1403). 여기에서, 마스킹은 상기 어드레스에 상기 "3(0000011)"을 반전시킨 값 "1111100"을 AND 연산하는 것에 의해 수행될 수 있다.
- <109> 상기 어드레스의 하위 2비트를 제외한 마스킹 되지 않은 나머지 비트들로부터 구간 정보를 얻는다(1404). 여기에서, 상기 구간 정보는 구간 순서( $i_0$ )이다.
- <110> 이 후, 상기 구간의 구간 번호( $i_N$ )가 "0"인지의 여부가 판단된다(1405). 상기 구간 번호( $i_N$ )가 "0"인 경우 상기 어드레스로부터 X0 값이 얻어진다(1406). 예를 들어, 상기 어드레스가 "5(101)"인 경우 X0 값은 상기 어드레스와 동일하게 "5(101)"가 된다.
- <111> 또, 구간 번호( $i_N$ )가 "0"이 아닌 경우 상기 스텝 정보에 소정의 값이 논리 연산되며(1407), 상기 논리 연산된 스텝 정보가 소정 횟수에 도달될 때까지 좌측으로 한 비트씩 쉬프트 된다(1408, 1409). 상기 소정의 값은 "4(100)"이고, 상기 쉬프팅 횟수는 구간 순서( $i_0$ ) - 1회 이다. 상기 논리 연산은 OR 연산으로 수행될 수 있다.

- <112> 이 후, 상기 소정 횟수만큼 쉬프팅이 완료되면, 상기 쉬프팅된 변화 비트(A)로부터 X의 근접 X좌표 값이 얻어진다(1410).
- <113> 예를 들어, 상기 변화 비트가 "01"이고, 구간 순서( $i_0$ )가 "2"일 때, 상기 변화 비트 "01"에 "4(100)"를 OR 연산하면 "101"이 되고, 이것을 1회 좌측으로 쉬프트 시키면 "1010"이 된다. 따라서, X0 값은 "1010"이 된다.
- <114> 상기 어드레스로부터 X0 값을 산출하는 과정을 C 언어로 표현하면 아래와 같다.
- <115> 【표 5】

```
int index2x(int index){
    int x = index & 0x3; /* 어드레스의 하위 2비트 */
    int i = index & ~0x3; /* 그 외의 상위 비트들 */
    if (i) {
        x |= 0x4;
        x <<= i-1;
    }
    return x;
}
```

- <116> 상기 X0 값의 산출 과정을 일예를 들어 좀 더 상세히 설명하면 다음과 같다.
- <117> 어드레스가 13(1101)인 경우, 하위 2비트를 제외한 나머지 비트들을 마스킹하면, 변화 비트 "01"만이 남게 된다. 또, 상기 어드레스 "1101"을 하위 2비트만을 마스킹 하면 상위 2비트인 "11"만이 남게 된다.
- <118> 따라서, 상기 상위 2비트가 "(3)11"이므로 구간 순서( $i_0$ )가 3인 것을 알 수 있다. 여기서, 상기 구간( $i_3$ )은 구간 번호( $i_N$ )가 0이 아니므로, 상기 변화 비트 "01"에 "(4)100"을 OR 연산하면, "101"이 된다.

<119>       여기에서, 상기 "101"을 구간 순서( $i_0$ )-1회, 즉, 2회 좌측으로 쉬프트 시키면, "10100"이 된다. 따라서, X0 값이 "10100"이 됨을 알 수 있다.

<120>       본 발명은 도면에 도시된 실시예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의 지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시예가 가능하다는 점을 이해할 것이다. 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 등록청구범위의 기술적 사상에 의해 정해져야 할 것이다.

#### 【발명의 효과】

<121>       상기한 것과 같이, 본 발명의 룩업 테이블 압축방법 및 이를 이용하여 압축된 룩업 테이블을 가지는 비선형 함수 발생장치 및 그 발생방법에 의하면, 룩업 테이블이 차지하는 메모리의 용량을 줄일 수 있는 효과가 있다.

<122>       또, 본 발명의 비선형 함수 발생장치에 의하면, 근사값 계산에 필요한 데이터들을 곱셈기를 사용하지 않고 쉬프트 연산만으로 계산하므로, 디지털 신호처리 반도체 장치의 크기를 줄일 수 있는 효과가 있다.

**【특허청구범위】****【청구항 1】**

기울기 변화가 감소하는 비선형 함수 발생장치의 록업 테이블 압축방법에 있어서,

- (a) 비선형 함수의 X좌표들을 소정의 좌표 간격을 가지는 스텝을 각각 포함하는 복수개의 구간들로 분할하는 단계;
- (b) 상기 각 스텝별 X좌표 값에 대응하는 Y좌표 값을 추출하는 단계; 및
- (c) 상기 Y좌표 값들을 메모리 내의 지정된 주소에 저장하는 단계를 포함하며,  
상기 스텝의 상기 좌표 간격은 상기 복수개의 구간들에 대하여 다르게 설정되는 것을 특징으로 하는 메모리를 감소시키는 개선된 록업 테이블 압축방법.

**【청구항 2】**

제1항에 있어서, 상기 복수개의 구간들 각각은,  
적어도 하나 이상의 스텝을 포함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 록업 테이블 압축방법.

**【청구항 3】**

제2항에 있어서, 상기 복수개의 구간들 각각은,  
2의 지수승의 스텝 수를 포함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 록업 테이블 압축방법.

**【청구항 4】**

제2항에 있어서,

상기 복수개의 구간들 각각에 포함되는 스텝들의 X좌표 값들은 이진수로 표현되고

상기 스텝의 이진수 중 하나 이상의 비트를 포함하는 변화 비트는 상기 X좌표 값이 속하는 구간 내에서 상기 X좌표 값에 대응하는 스텝 정보를 산출하는데 이용되는 것을 특징으로 하는 메모리를 감소시키는 개선된 룩업 테이블 압축방법.

【청구항 5】

제1항에 있어서,

상기 복수개의 구간들 중 일부 구간들은 서로 동일한 좌표 간격의 스텝을 포함하며,

상기 복수개의 구간들 중 나머지 구간들은 서로 다른 좌표 간격의 스텝을 포함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 룩업 테이블 압축방법.

【청구항 6】

제5항에 있어서, 상기 나머지 구간들은,

상기 비선형 함수의 기울기 변화가 감소되는 구간일 수록 스텝의 좌표 간격이 증가되는 것을 특징으로 하는 메모리를 감소시키는 개선된 룩업 테이블 압축방법.

【청구항 7】

제6항에 있어서, 상기 스텝의 좌표 간격은,

2의 지수승으로 증가되는 것을 특징으로 하는 메모리를 감소시키는 개선된 룩업 테이블 압축방법.

## 【청구항 8】

기울기 변화가 감소하는 비선형 함수를 발생시키는 장치에 있어서,  
 입력되는 X좌표 값으로부터 어드레스를 포함하는 분석정보를 산출하여 출력하는  
 분석기;

소정의 좌표 간격을 가지는 스텝을 포함하는 복수개의 구간들로 분할되는 비선형  
 함수의 상기 스텝별 X좌표 값에 대응하는 Y좌표 값이 지정된 주소에 저장되는 록업 테이블  
 을 포함하며, 상기 어드레스에 대응하는 Y좌표 값을 출력하는 메모리; 및

상기 분석정보 및 상기 어드레스에 대응하는 Y좌표 값들로부터 상기 입력되는 X좌  
 표 값에 대응하는 Y좌표의 근사값을 계산하여 출력하는 연산기를 구비하며,

상기 스텝의 상기 좌표 간격은 상기 복수개의 구간들에 대하여 다르게 설정되는 것  
 을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

## 【청구항 9】

제8항에 있어서, 상기 분석정보는,

상기 입력되는 X좌표 값이 속하는 구간의 구간정보 및 스텝정보와, 상기 스텝의 좌  
 표 간격, 상기 입력되는 X의 근접 X좌표 값을 포함하는 것을 특징으로 하는 메모리를 감  
 소시키는 개선된 비선형 함수 발생장치.

## 【청구항 10】

제8항에 있어서, 상기 어드레스는,

상기 입력되는 X좌표가 속하는 스텝의 어드레스 및 그 다음 스텝의 어드레스를 포  
 함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

**【청구항 11】**

제8항에 있어서, 상기 복수개의 구간들 각각은,

적어도 하나 이상의 스텝을 포함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

**【청구항 12】**

제11항에 있어서, 상기 복수개의 구간들 각각은,

2의 지수승의 스텝 수를 포함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

**【청구항 13】**

제8항에 있어서,

상기 복수개의 구간들 중 일부 구간들은 서로 동일한 좌표 간격의 스텝을 포함하며,

상기 복수개의 구간들 중 나머지 구간들은 서로 다른 좌표 간격의 스텝을 포함하는 것을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

**【청구항 14】**

제8항에 있어서, 상기 나머지 구간들은,

상기 비선형 함수의 기울기 변화가 감소되는 구간일 수록 스텝의 좌표 간격이 증가되는 것을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

**【청구항 15】**

제8항에 있어서, 상기 스텝의 좌표 간격은,

2의 지수승으로 증가되는 것을 특징으로 하는 메모리를 감소시키는 개선된 비선형 함수 발생장치.

【청구항 16】

기울기 변화가 감소하고, 소정의 좌표 간격을 가지는 스텝을 포함하는 복수개의 구간들로 분할되는 비선형 함수의 상기 스텝별 X좌표 값에 대응하는 Y좌표 값이 지정된 주소에 저장되는 룩업 테이블을 포함하는 비선형 함수 발생장치의 비선형 함수 발생방법에 있어서,

- (a) 입력되는 X좌표 값으로부터 구간 정보 및 스텝 정보를 산출하는 단계;
- (b) 상기 구간 정보 및 스텝 정보로부터 어드레스를 산출하는 단계;
- (c) 상기 룩업 테이블로부터 상기 어드레스에 대응하는 Y좌표를 획득하는 단계;
- (d) 상기 어드레스로부터 입력되는 X좌표의 근접 X좌표 값을 산출하는 단계;
- (e) 상기 구간정보로부터 해당 구간에 포함되는 스텝의 좌표 간격 값을 산출하는 단계; 및
- (f) 상기 산출된 X좌표의 근접 X좌표 값, 상기 어드레스에 대응하는 Y좌표 값 및 상기 좌표 간격 값으로부터 상기 입력 X좌표 값에 대응하는 Y좌표의 근사값을 계산하여 출력하는 단계를 포함하는 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 17】

제16항에 있어서,

상기 복수개의 구간들 각각에 포함되는 스텝들의 X좌표 값들은 이진수로 표현되고

상기 스텝의 이진수 중 하나 이상의 비트를 포함하는 변화 비트는 상기 X좌표 값이 속하는 구간 내에서 상기 X좌표 값에 대응하는 스텝 정보를 산출하는데 이용되는 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 18】

제16항에 있어서, 상기 (a) 단계는,

(g) 소정의 이진 비트들을 가지는 X좌표 값을 수신하는 단계;

(h) 소정 값 미만이 될 때까지 상기 X좌표 값의 각 비트들을 우측으로 한 비트씩 쉬프트시켜 최종 쉬프팅 값을 얻는 단계;

(i) 상기 X좌표 값의 쉬프팅 횟수를 카운팅하는 단계;

(j) 상기 쉬프팅 횟수로부터 상기 구간정보를 얻는 단계; 및

(k) 상기 최종 쉬프팅 값으로부터 스텝정보를 얻는 단계를 포함하는 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 19】

제18항에 있어서, 상기 소정 값은,

한 구간 내에 포함되는 스텝의 수의 2배의 값인 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 20】

제18항에 있어서,

상기 구간정보는 구간 번호를 포함하며,

상기 스텝정보는 구간 내의 스텝의 위치인 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 21】

제20항에 있어서, 상기 (b) 단계는,

(1) 상기 구간정보를 나타내는 이진 비트들을 좌측으로 한 비트씩 소정 횟수만큼 쉬프트시키는 단계; 및

(m) 상기 쉬프트된 구간정보에 상기 스텝정보를 가산하여 상기 어드레스를 산출하는 단계를 포함하는 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 22】

제21항에 있어서, 상기 쉬프팅 횟수는,

상기 스텝의 수가  $2^t$ 일 때,  $t$ ( $t$ 는 자연수)회인 것을 특징으로 하는 비선형 함수 발생방법.

【청구항 23】

제17항에 있어서, 상기 (d) 단계는,

(n) 상기 어드레스를 나타내는 이진수 중 소정의 하위 비트 그룹으로부터 상기 변화 비트를 얻는 단계;

(o) 상기 어드레스를 나타내는 이진수 중 소정의 상위 비트 그룹으로부터 구간 정보를 얻는 단계; 및

(p) 상기 구간 정보에 따라 상기 변화 비트로부터 상기 X좌표의 근접 X좌표 값을 얻는 단계를 포함하는 것을 특징으로 하는 비선형 함수 발생방법.

**【청구항 24】**

제23항에 있어서, 상기 구간 정보는,

구간 순서와 구간 번호를 포함하는 것을 특징으로 하는 비선형 함수 발생방법.

**【청구항 25】**

제24항에 있어서, 상기 (p) 단계는,

상기 구간 번호가 '0'인 구간일 때, 상기 변화 비트가 상기 X좌표의 근접 X좌표 값인 것을 특징으로 하는 비선형 함수 발생방법.

**【청구항 26】**

제24항에 있어서, 상기 (p) 단계는,

(q) 상기 구간 번호가 '0'이 아닌 구간일 때, 상기 변화 비트에 소정의 데이터를 논리 연산하는 단계; 및

(r) 상기 논리 연산된 변화 비트를 소정 횟수만큼 좌측으로 쉬프트시켜 X좌표를 얻는 단계를 포함하는 것을 특징으로 하는 비선형 함수 발생방법.

**【청구항 27】**

제26항에 있어서, 상기 소정의 데이터는,

한 구간의 스텝 수인 것을 특징으로 하는 비선형 함수 발생방법.

**【청구항 28】**

제26항에 있어서, 상기 논리 연산은,

OR 연산인 것을 특징으로 하는 비선형 함수 발생방법.

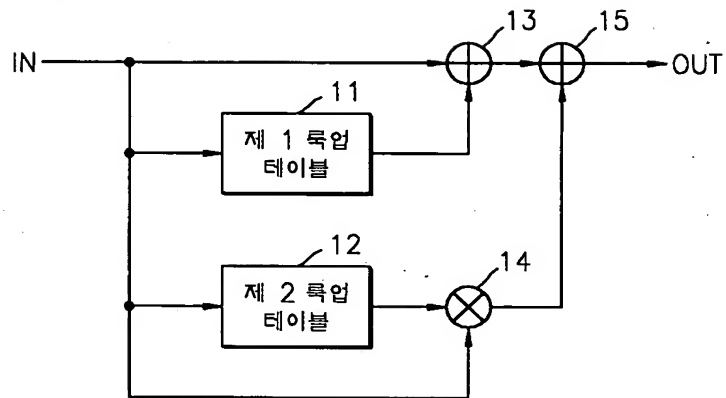
【청구항 29】

제26항에 있어서, 상기 쉬프팅 횟수는,

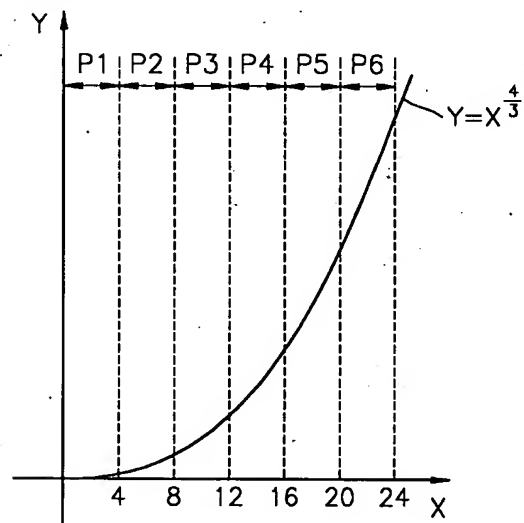
상기 구간 번호 보다 1 작은 것을 특징으로 하는 비선형 함수 발생방법.

【도면】

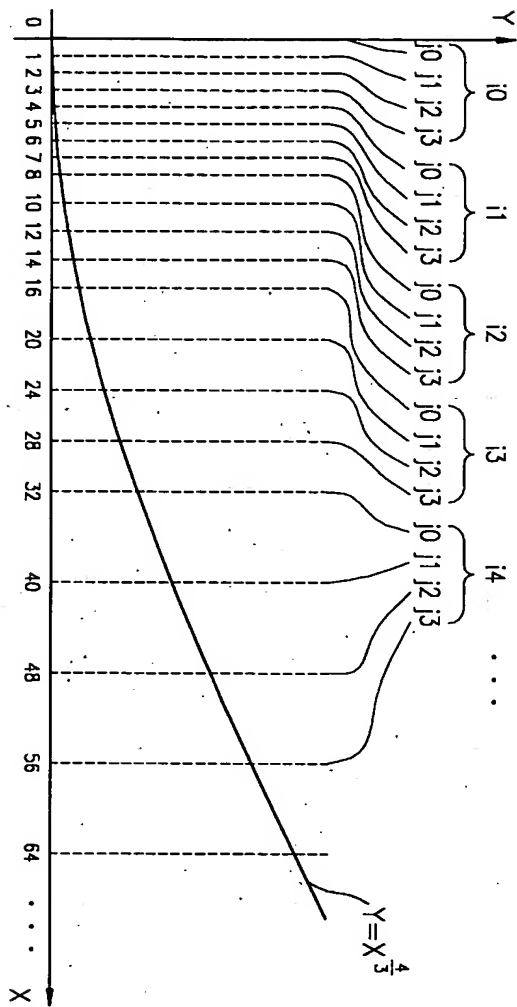
【도 1】



【도 2】



【도 3】



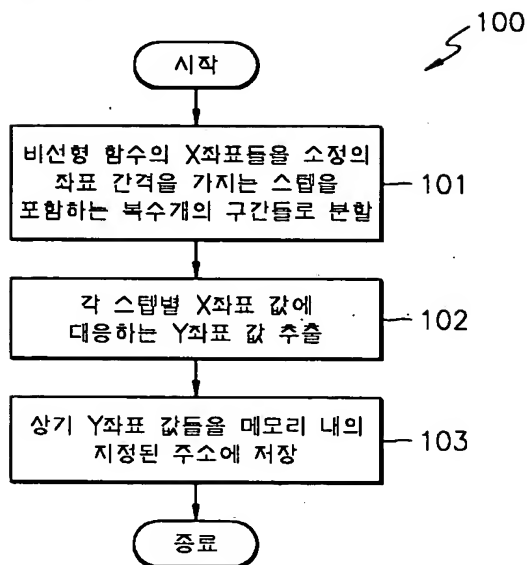


【도 4】

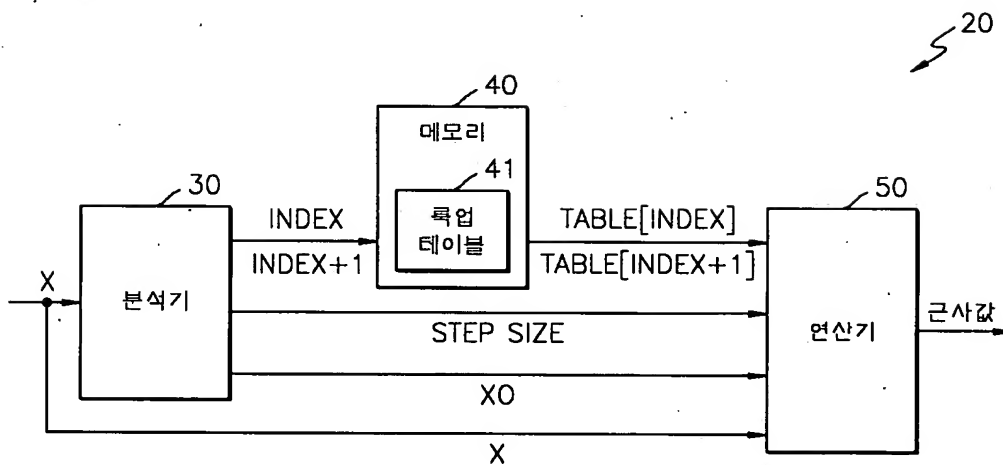
41

구간(i)		스텝(j)	X 좌표		Y 좌표	어드레스
구간순서 (iO)	구간번호 (iN)		이진수	십진수		
i0	0	0	00000000	0	0.00000	0
		1	00000001	1	0.00001	1
		2	00000010	2	0.00002	2
		3	00000011	3	0.00003	3
i1	0	0	00001000	4	0.00004	4
		1	00001001	5	0.00005	5
		2	00001010	6	0.00007	6
		3	00001011	7	0.00008	7
i2	.1	0	00010000	8	0.00010	8
		1	00010010	10	0.00013	9
		2	00010100	12	0.00017	10
		3	00010110	14	0.00020	11
i3	2	0	00100000	16	0.00024	12
		1	00100100	20	0.00033	13
		2	00101000	24	0.00042	14
		3	00101100	28	0.00052	15
i4	3	0	01000000	32	0.00062	16
		1	01001000	40	0.00083	17
		2	01010000	48	0.00106	18
		3	01011000	56	0.00130	19

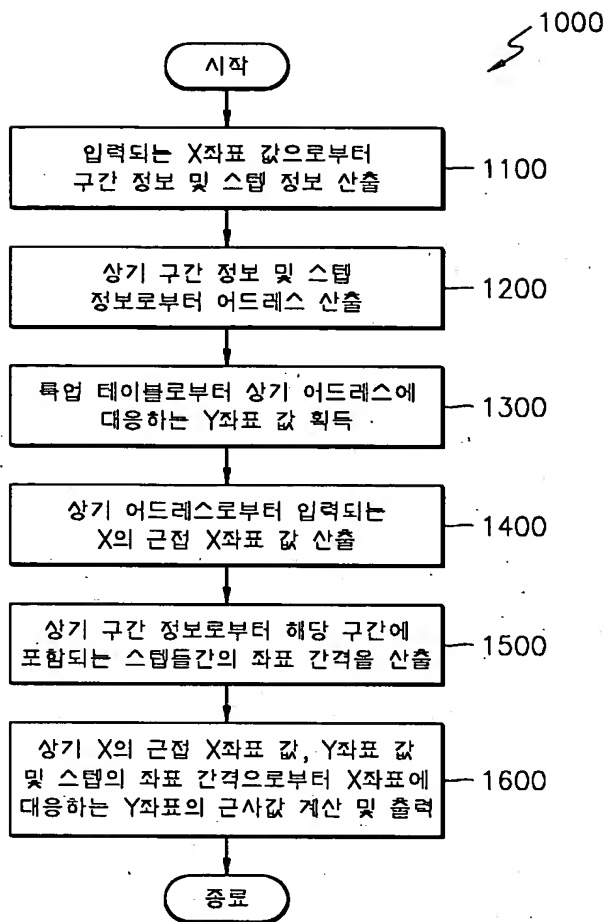
【도 5】



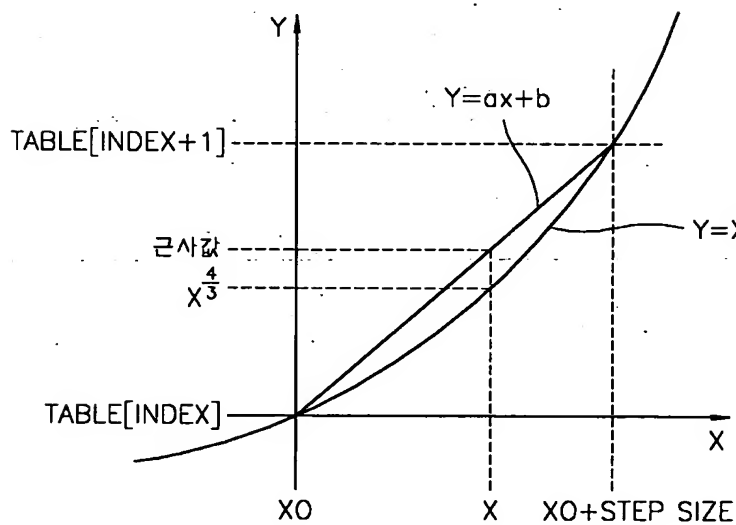
【도 6】



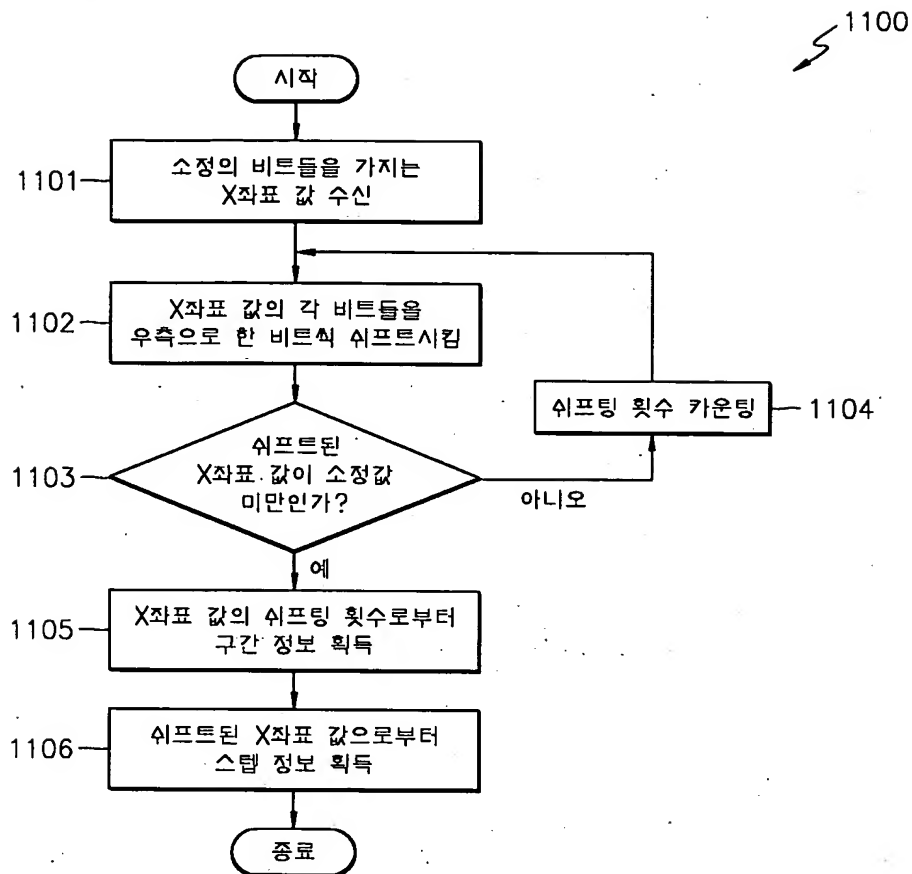
【도 7】



【도 8】



【도 9】



【도 10】

